5D－1

分野特化CASEにおける支援ツールの開発プロセスについて[1]

○篠崎政久　藤巻昇　小尾俊之[2]

株式会社　東芝　研究・開発センター
システム・ソフトウエア生産技術研究所[3]

1. はじめに

1. 1. 分野特化 CASE の有意性

我々は今までに支援対象を特定の製品に特化した分野特化 CASE ツールの開発・適用を行ってきた[1].

分野特化 CASE では十分に抽象化された仕様記述言語を用いて仕様記述を行い，その仕様から完全なプログラムの自動生成・検証等を行うことで，きめの細かい，一貫した開発支援を行うことが可能となり，より高い開発効率が実現されている.

1. 2. 分野特化 CASE 構築の問題点

分野特化 CASE を構築する際には 1.1 で述べた様なその分野に適した開発支援の実現を目的としなければならない.

しかし，誤った開発プロセスで分野特化 CASE を構築すると，要求仕様を十分に記述できなかったり，完全なプログラムの自動生成が不可能になる等，一貫した開発支援が実現できなくなってしまう．また，誤った開発プロセスで開発された CASE ツールは冗長な部分が発生しやすく十分な開発効率が得られなくなる場合がある.

このような状況は分野特化 CASE の開発プロセスが定義されていないことに起因する．また，開発プロセスの定義がなされていないことは，分野特化 CASE の開発において無駄や後戻が発生しやすく，多くの時間を取られてしまうことになる.

そこで，本稿では前で述べたような不具合が発生せず，また，十分な開発効率が得られるような分野特化 CASE の開発プロセスを提案する.

2. 分野特化 CASE の開発プロセス

図 2-1 に今回提案する分野特化 CASE の開発プロセスを示す．各段階では前段階での成果を元に定義を進めて行く．但し，前段階より前の成果から影響を受ける場合はその影響も考慮に入れて，定義を行う必要がある.

また，定義を行う際には下位の段階から上位の段階へ影響を及ぼさないように行なう必要がある．そのような影響を与えるとアーキテクチャに基づいた開発の支援を行えなくなる.

次より各定義の詳細を述べて行く.

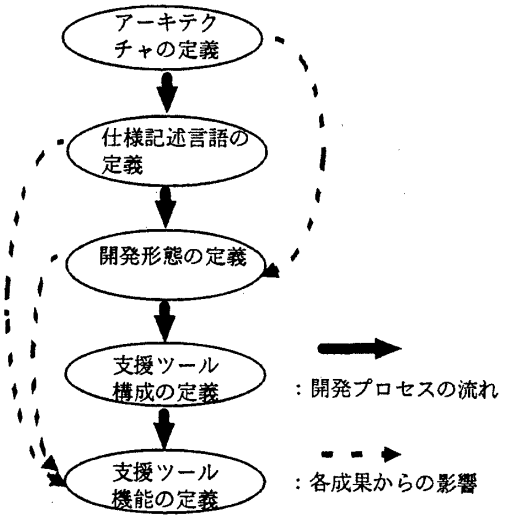

図 2.-1　分野特化 CASE ツールの開発プロセス

2. 1. アーキテクチャの定義

ソフトウエア開発を行う際には最初に要求を分析し，ソフトウエアのアーキテクチャ（骨組み）を定義する．次にそのアーキテクチャにそってプログラムを開発し，目的とするソフトウエアを作成する．つまり，アーキテクチャはプログラム開発の基本となるものである．分野特化 CASE を用いた開発においてもこのアーキテクチャに基づいたプログラム開発を実現することが目的である.

そのためアーキテクチャの定義は対象となる分野の特徴をとらえ，基本となる機能とカスタマイズ可能な部分等を考慮にいれながら行う必要がある.

2. 2. 仕様記述文法の定義

仕様記述文法の定義は，クラス図や状態遷移図等の標準的な記述方法をアーキテクチャモデルにあわせてカスタマイズを行ったり，追加を行って定義していく．その際には要求仕様を仕様上に完全に記述できるように記述実験を繰り返しながら定義してい

---

[1] A development tools' process on domain special CASE
[2] Masahisa Shinozaki,Noboru Fujimaki and Toshiyuki Obi
[3] TOSHIBA Corporation R&D Center, Systems & Software Engineering Laboratory

かなければならない．そのため，変更容易性が求められる．また，記述の容易さや，レビューがしやすい様に理解性の高い，十分に抽象化を行った仕様記述文法を定義する必要がある [2].

2. 3. 開発形態の定義

開発形態の定義は一貫した開発支援が行え，十分な開発効率を得られるように行う必要がある．また，その開発形態によってアーキテクチャに基づいた開発支援が可能でなければならない．

このような開発形態は次の 3 段階から導き出すことができる．

(1) 開発形態の特徴を挙げる．
(2) 作業内容と成果物との間連を定義する．
(3) 作業の順番・分担を定義する．

(1)であるが，オブジェクト指向開発や分散開発など，開発の特徴を挙げ，開発形態の概要を決定する．

(2)では作成すべき仕様書などの成果物と作業内容を洗い出し，成果物と作業の関係を決定する．

(3)で成果物と作業の関係，作業の並行性などを考慮し，作業の流れや分担を決定する．必要があれば(2)へ戻る．

オブジェクト指向分析によって構築され，フレームワーク（標準的な仕様）を持つアーキテクチャからこのようにして定義された開発形態の一例を DFD で図 2.-2 に挙げる．図の中の番号は作業順序を示す．

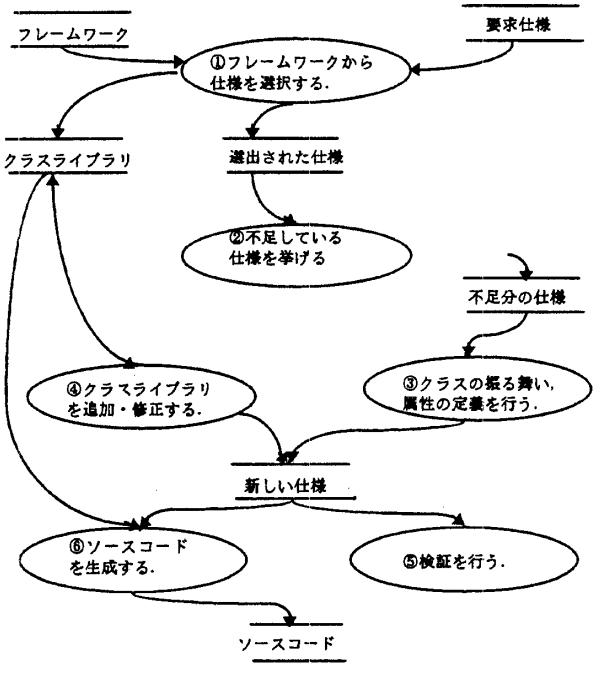

図 2.-2 開発形態の例

2. 4. 支援ツール構成の定義

支援ツール構成の定義は，定義された開発形態の中からツール化の効果のある作業，あるいは，ツールが必須となる作業を選出することによって決定される．例えば図 3-1 の場合，次のようなツール構成が考えられる．クラスセレクター（作業①），仕様記述エディタ，文法チェッカ，クラスブラウザ（作業③），シミュレータ（作業⑤），ジェネレータ（作業⑥）．ツール構成の定義を行う際には一貫した支援が可能であるかを十分吟味し，決定する必要がある．

2. 5. 支援ツール機能の定義

支援ツール機能の定義を行う際にも一貫した支援を考慮しながら，各ツールの機能を定義しなければならない．また，機能決定の際には既に行われた仕様記述言語の定義や開発形態の定義からの影響も考慮に入れて定義を行わなければならない．

ここで，開発のプラットフォーム的役割を持つ仕様記述エディタを例に「ツールによる開発手順のコントロール」と「仕様記述文法を考慮に入れたツールの実現法」示す．

分野特化 CASE は定義された開発形態にそって開発を行う時，もっとも高い開発効率が得られる．ユーザをその正しい開発の流れに導くため，エディタは開発形態に沿った各種ツールの起動を行うことを可能とし，ツールの起動に制限を設ける．更に，仕様記述の順番を制限するなどして仕様の抜けを防止することができる．

また，仕様記述エディタは仕様記述言語で定められている記号等を表現できなければならない．また，文法の特徴を捕らえることによりユーザが容易に仕様記述が行え，ミスが少なくなるようなオペレーションやチェック機能を備えていなければならない．

3. おわりに

本稿では分野に特化した CASE ツールを開発する際のプロセスを提案した．この開発プロセスによって作成された分野特化 CASE はアーキテクチャから導びかれるため，アーキテクチャに基づいた開発を支援できる CASE となる．また，各定義の方法とその定義の際に考慮に入れなければならない定義と絶対条件・指針を示したことにより分野特化 CASE 構築の効率化を図ることができるしかし，今回提案した開発プロセスは各段階の独立性が低いため前段階の定義に変更が生じた時の影響が大きく，分散開発に向いていないという欠点がある．今後は各段階の独立性を高め，より効率的な開発プロセスとしていく．また，各段階の定義方法はまだ試行錯誤によるため，定義に時間がかかってしまう．各段階の定義方法も形式化，プロセス化を行うことにより，効率の良い CASE ツール開発を行うことができる．

[1] 清水他，“金融機器組み込みソフト向け CASE”，情報処理学会研究会報告 93-SE-91,, 1993
[2] 若松直樹，玉木裕二，清水洋子，小尾俊之，“形式的仕様記述におけるわかりやすいネーミング”，第 53 回全国大会公演論文集